Australian Government
Department of Defence
Defence Science and
Technology Organisation

Experience with a System for Manual Clustering of Air Surveillance Track Data

Matthew C. Lowry

Command, Control, Communications and Intelligence Division
Defence Science and Technology Organisation

DSTO-GD-0749

ABSTRACT

Using clustering algorithms to mine air surveillance track data for groups of similar flights has the potential to facilitate a variety of capability enhancements. Since there are many algorithms that could be used, a method for assessing the quality of algorithm output is required. One potential method is to have a human expert hand-craft a clustering for a test data set, and use this manual clustering as the gold-standard against which the output of a clustering algorithm is assessed. For complex spatio-temporal data such as air surveillance track data, the manual construction of clusterings for a robust test data suite will be labour-intensive and reliant on good information technology support. This report describes an experimental system providing a user interface and workflow for performing manual clustering of air surveillance track data, and experience with a trial of the system.

RELEASE LIMITATION

Approved for public release

Published by

Command, Control, Communications and Intelligence Division
DSTO Defence Science and Technology Organisation
PO Box 1500
Edinburgh South Australia 5111 Australia

Telephone: 1300 DEFENCE
Fax: (08) 7389 6567

AR-015-637
April 2013

APPROVED FOR PUBLIC RELEASE

Experience with a System for Manual Clustering of Air Surveillance Track Data

Executive Summary

Clustering is a data mining technique for analysing large data sets and finding groups within the data that are in some way close or similar to each other. Clustering algorithms seek to produce high-quality clusterings. For a clustering to have high quality, not only must similar data elements be placed in the same group, but also data elements that are dissimilar must not be placed in the same group.

Clustering algorithms offer a means to exploit archives of air surveillance track data. Applying clustering algorithms to such data can identify groups of flights that occurred at different points in time but exhibited similar flight paths and behaviour. This form of analysis could enable improved capability in a variety fields including situational awareness and tactical air intelligence (automated behaviour prediction and anomaly detection, indicators and warnings, *etc.*), strategic air intelligence (historical analysis, capability assessment, *etc.*), and general efficiency dividends (higher performance of air surveillance and air intelligence operators, improved training and knowledge retention practices, *etc.*).

Realising the potential benefits of discovering clusters within air surveillance track data will depend heavily upon obtaining high-quality results from clustering algorithms. This leads to the problem of how to make an objective, quantitative assessment of the quality of clustering algorithm output.

Examining the issues surrounding the assessment of the quality of clustering algorithm output is beyond the scope of this report (there is a prior technical report that contains a more general discussion of that topic). The scope of this report is a single issue associated with clustering quality assessment; namely the practical issue of constructing a hand-crafted ideal against which the output of an algorithm can be compared. This is a labour intensive process that cannot be performed on a large, complex data set such as an archive of air track data without appropriate information technology support. This report describes a prototype system that provides such support, and experience with a trial of the system.

This page intentionally left blank

Contents

This page intentionally left blank

1. Introduction

Clustering is a data mining technique for analysing large data sets and finding groups within the data that are in some way close or similar to each other. Clustering algorithms seek to produce high-quality clusterings. For a clustering to have high quality, not only must similar data elements be placed in the same group, but also data elements that are dissimilar must not be placed in the same group.

A comprehensive introduction to clustering is beyond the scope of this report; the reader is directed to a popular textbook by Han and Kamber (2006) for an introduction and an article by Jain, Murty and Flynn (1999) for a comprehensive survey of clustering techniques.

Clustering algorithms offer a means to exploit archives of air surveillance track data. Applying clustering algorithms to such data enables the discovery of groups of flights that occurred at different points in time but exhibited similar behaviour and followed similar flight paths. This form of analysis could enable improved capability in a variety fields including:

- situational awareness and tactical air intelligence (automated behaviour prediction and anomaly detection, indicators and warnings, *etc.*)
- strategic air intelligence (historical analysis, capability assessment, *etc.*)
- general efficiency dividends (higher performance of air surveillance and air intelligence operators, improved training and knowledge retention practices, *etc.*).

Realising the potential benefits of discovering clusters within air surveillance track data will depend heavily upon obtaining high-quality results from clustering algorithms. This leads to the problem of how to make an objective, quantitative assessment of the quality of clustering algorithm output.

1.1 Clustering Quality Assessment

In general the academic literature recognises two basic approaches to performing clustering quality assessment (Jain, Murty and Flynn 1999).

Internal quality assessment involves developing a function that applies directly to a clustering result and yields a number to describe the quality of the clustering. The function examines the structure of the clustering, the distances between data elements and the choices made by the clustering algorithm to group or not group together elements, and makes a quality assessment based on some abstract notion of the structure that an ideal clustering result would have.

External quality assessment involves developing a hand-crafted ideal clustering result. A manually constructed "gold standard" clustering can be taken as defining the perfect

output for a clustering algorithm to generate. Then the actual output of any clustering algorithm can be assessed for quality by measuring how similar it is to the ideal.

Examining the issues surrounding these two approaches is beyond the scope of this report; there is a prior technical report (Lowry 2013) that contains a more general discussion of the topic. The scope of this report is a single issue associated with external quality assessment; namely the practical issue of constructing the hand-crafted ideal against which the output of an algorithm can be compared. This is a labour intensive process that cannot be performed on a large, complex data set such as an archive of air track data without appropriate information technology support.

This report describes a prototype system that provides such support, and experience with a trial of the system. The system in question is briefly described below in Section 1.2. The user interface and supported workflow for manual track clustering is described in Section 2. Details of a trial of the system and lessons observed are related in Section 3.

1.2 Blini Software

The Blini[1] system is a software package developed by the author of this report to investigate the issues surrounding the exploitation of air surveillance track data via clustering algorithms. The graphical user interface for manual clustering of air surveillance track data described in this report was implemented within the Blini system. A brief overview of the system is given here as context for the subsequent discussion.

The functional requirements the Blini system was designed to address are:

a) Store corpora of track data for experimental purposes.

b) Provide a graphical user interface (GUI) in which the user can visually explore the track data.

c) Store clusterings describing structure in the corpora of track data.

d) Provide a GUI in which the user can visually explore the clusterings of the track data.

e) Implement clustering algorithms; allow execution of clustering algorithms over the track data to generate clusterings.

f) Implement clustering quality functions for internal quality assessment; evaluate quality functions on stored clusterings.

g) Provide a graphical user interface in which the user can manually construct a clustering on the track data.

1. *Blini* [plural; from Russian блины (bliný), singular блин (blin)]: thin Russian pancakes similar in consistency to the French *crêpe* but made with eggless yeast-risen batter, and often served with savoury garnishes. Using wheat flour is typical in modern times, but traditionally they were made with buckwheat flour and were more akin to the *galette bretonne* of French cuisine.

h) Implement clustering similarity functions for external quality assessment; evaluate similarity functions between hand-crafted and algorithmically-generated clusterings.

Blini is implemented with a client/server architecture. The storage of track data and clusterings is handled by the server component, which is simply an off-the-shelf relational database system. The algorithm execution and GUI aspects of the system are implemented in the client component, which is a Java application.

This report discusses only how requirement (g) above was addressed in Blini. Two points of relevance to the subsequent discussion are:

- The GUI is implemented using the features of the standard Swing/AWT libraries from recent versions of the Standard Edition of the Java platform. Thus the design and behaviour of the GUI is constrained by the features of the Swing/AWT libraries.
- The GUI was developed as an extension from the interface components initially developed for requirements (b) and (d) above. Thus the design and behaviour of the GUI reflects to some extent the goal of rapid implementation via code and component re-use, rather than good GUI design *per se*.

2. Manual Track Clustering User Interface and Workflow

The user interface provided by Blini is designed to support an iterative process for manual clustering construction.

The process commences with the user selecting a corpus of track data. Initially all tracks in the corpus are considered "unallocated". Each iteration of the construction process performed by the user will result in either:

a) a cluster being created and a group of tracks allocated to the cluster, or

b) one or more tracks being declared an outlier (*i.e.* a track that does not belong in a group with other tracks).

Eventually there will be no unallocated tracks remaining, and the manual construction of the clustering is complete.

2.1 Main Interface Window

The main interface presented to the user for performing this process is shown in Figure 2 (page 14). The interface is divided into three vertical panes.

a) The left-hand side of the interface contains the clusters pane. This pane lists the clusters that have been created so far. As the user makes progress in allocating tracks to clusters, they will observe this list growing. The user can re-inspect

and modify clusters they have already created by selecting a cluster listed in this pane.

b) The centre pane contains controls for the user to operate on unallocated tracks. The two operations that the user can perform on unallocated tracks are allocating them to a cluster or declaring them to be outliers. Correspondingly there are two distinct modes of operation:

- "*Search from seed*", in which the user assembles a collection of unallocated tracks that are close to each other and should form a cluster.
- "*Auto outlier marking*", in which the system suggests a list of tracks that appear to be outliers, and the user examines the tracks to confirm their outlier status.

These two modes are described in more detail in the subsequent sections.

c) The right-hand side pane contains a simple geographic map display. As the user works to construct clusters or declare tracks as outliers, the map display will shows the tracks and clusters the user is working on within their geographic context. The map contains basic geographic features that are relevant to analysing the air surveillance track data (*e.g.* coastlines, cities, aerodromes, *etc.*). Further, the map display supports animated "playback" of tracks, showing the dynamic behaviour over time of the tracks being displayed.

2.2 Search from Seed

The "Search from seed" mode is the mechanism by which the user constructs new clusters. The procedure is as follows:

1) The system selects (arbitrarily) a track from the pool of unallocated tracks. This track will be used as the "seed" for a search within the remaining unallocated track pool for other tracks that are close to the seed.
2) The user selects a track separation measure and a separation threshold.
3) The system searches for any track that is within the user-specified threshold distance of the seed, as assessed by the user-specified track separation measure. Any track within the threshold is considered a candidate for inclusion in the cluster and placed in a list which is presented to the user. The list is sorted by distance from the seed track, allowing the user to rapidly inspect marginal cases - *i.e.* the tracks that almost fell outside the threshold.
4) The user removes from the list any track deemed inappropriate for inclusion in the cluster. At any point the user may return to step 2 (*i.e.* adjust the separation measure and threshold and repeat the search with the updated search settings).
5) Once the user is satisfied that a high-quality cluster will be formed by the tracks remaining in the list they instruct the system to form a cluster. A new cluster is

created; the tracks in the list are allocated to the new cluster and removed from the unallocated pool.

6) Alternatively, if the user determines that the number of tracks close to the seed track is insufficient to form a cluster, the user can declare the seed track to be an outlier.

By repeating this with another seed from the unallocated track pool, the user will eventually cause all tracks to be either allocated to clusters or declared as outliers.

2.2.1 Demonstration

A demonstration of this process is depicted in Figures 3, 4, and 5 (pages 15, 16, and 17).

Figure 3 (page 15) shows the interface immediately after the user has instructed the system to select the next seed track and make a search for other nearby tracks using the specified track separation measure and threshold. The seed track is displayed as a yellow flight path on the map display. The other tracks in the database that have a similar flight path, and are within the separation threshold specified by the user, have also been placed in the list and are displayed on the map in green.

In the example in Figure 3, the seed track is a flight from Kalgoorlie-Boulder to Perth International Airport (marked on the map by its ICAO[2] location code YPPH). The search for unallocated tracks with a similar flight path to the seed has found ten additional tracks.

Amongst the similar tracks suggested by the system, there are eight flights between the same locations. Three distinct flight paths between the two locations are visible — some of the flights approach Perth from the south, some from the north, and some flights initially adopt the southern approach route but are redirected to the northern approach route. In addition, the system has identified two other flights that are within the threshold of closeness to the seed track. However from the map display it is clear these two flights do not originate near Kalgoorlie-Boulder, and are not following the same flight path.

For the sake of the example, assume the user's intuition is that the latter two tracks should be excluded from the cluster. To validate their decision, the user can observe that the two tracks have a significantly higher separation from the seed (around 125 km) compared to the former eight tracks (all have separation less than 75 km). Also, the user can have the system perform a "replay" of the tracks, as depicted in Figure 4 (page 16).

During a replay of the tracks in the map display, the system provides a pop-up window with a play/pause button and time slider. The time slider corresponds to time after takeoff for each track — so in the example of Figure 4, where the slider is at the

[2] International Civil Aviation Organization; an agency of the United Nations that standardises, amongst other things, four-letter location codes to identify aerodromes throughout the world.

"00:17:06" mark, the system is displaying the position of each track at 17 minutes and 6 seconds into that track's flight.

From a playback of the tracks, it is clear that the seed track and the other eight tracks between Kalgoorlie-Boulder and Perth are forming a cluster along the flight path between the two locations. It is also clear that the other two flights do not belong with this cluster.

The user removes the two tracks that do not belong with the others and forms a cluster from the remaining tracks. The result can be seen in Figure 5 (page 17). The system has created a new cluster containing nine tracks — the seed and the eight tracks close to the seed confirmed by the user. The system displays the newly created cluster on the map display. Note how the seed track does not have any special status in the cluster; it is merely one of the component tracks in the cluster. In Figure 5 the component tracks are displayed in cyan, and the system also displays a synthetic representative track which the system has computed by averaging the component tracks.

2.3 Automatic Outlier Marking

The primary means for the user to make progress in allocating tracks to clusters is to form new clusters with the procedure described previously. The system provides a second mechanism to augment that procedure. The automatic outlier marking procedure allows the user to rapidly handle outliers.

To employ this procedure, the user selects a track separation measure, a threshold for track separation, and also a number of tracks threshold. The system searches for any unallocated track where the quantity of other unallocated tracks within the specified distance is no more than the specified number. All tracks the system finds that fall within the user-specified thresholds for outlier status are placed in a list, and the user can inspect any of the proposed outliers.

By examining proposed outliers the user can assess whether the thresholds for the outlier search were sufficiently conservative to ensure only tracks that actually deserve outlier status have been added to the list. If the user encounters a track in the list that should not be marked an outlier, they can adjust the thresholds and make the outlier search more conservative. Once the user is confident that the list proposed by the system does not contain false positives, the user indicates to the system that it may mark all tracks in the list as outliers, and the system removes these tracks from the unallocated pool.

2.3.1 Demonstration

A demonstration of the automatic outlier detection process is depicted in Figures 6 and 7 (pages 18 and 19).

Figure 6 shows the user setting the thresholds; they have selected a separation threshold of 150 km and a nearby track threshold of one. So under these settings the system will be searching for any track where there is no more than one other track within a separation of 150 km.

Figure 7 shows the outcome after an outlier search. The system lists the tracks it has found, and gives the user the opportunity to inspect the tracks and validate their outlier status. The user can select one of the suggested outlier tracks, and the map display will show that track and any other close tracks that were detected.

In Figure 7 the user is inspecting a proposed outlier. The track, displayed in yellow on the map, originates from Jandakot Airport (location code YPJT — the primary light airport serving Perth), takes a "scenic route" to the vicinity of the township of Busselton, and after loitering in that area returns to Jandakot by the same route. The system has found another unallocated track that also originates from Jandakot airport, flies to the Busselton area, loiters, and returns. However this other track, displayed on the map in green for comparison, takes a more direct route. The user is able to inspect the two tracks and confirm that they are sufficiently different and hence the proposed outlier track should not be placed in a cluster with any other unallocated tracks.

3. Trial and Lessons Observed

A trial was performed to assess the efficiency of the workflow supported by the Blini system. The experience with the trial gave insight into the aspects of the interface and workflow that were effective, and how improvements might be made.

The trial was performed by the author of this report, and involved hand-crafting a clustering for a trial corpus of 5000 air surveillance tracks describing the real-world movement of aircraft. The data set covers the air space in and around Australia's Flight Information Region, over a period of approximately 36 hours from 2008-04-21 00:00 UTC to 2008-04-22 12:00 UTC. The tracks were taken from an unclassified subset of the *Recognised Air Picture* (RAP) generated by RAAF 41 Wing.

Rate of Progress

Table 1 shows the rate of progress made by the author allocating tracks to clusters during the trial. The workflow supported by the Blini system encourages the user to perform the task of allocating tracks to clusters in discrete chunks of work with clear break points between each chunk. The Blini system allows the user to stop work, and even terminate the application if needed, before recommencing.

Table 1: Progress allocating tracks during trial of manual cluster construction

Session duration	Total time elapsed	Tracks remaining
	0	5000
102	102	4440
60	162	4053
42	204	3715
60	264	3283
31	295	3052
30	325	2841
49	374	2618
50	424	2314
15	439	2243
79	518	1858
18	536	1752
57	593	1392
104	697	623
55	752	0

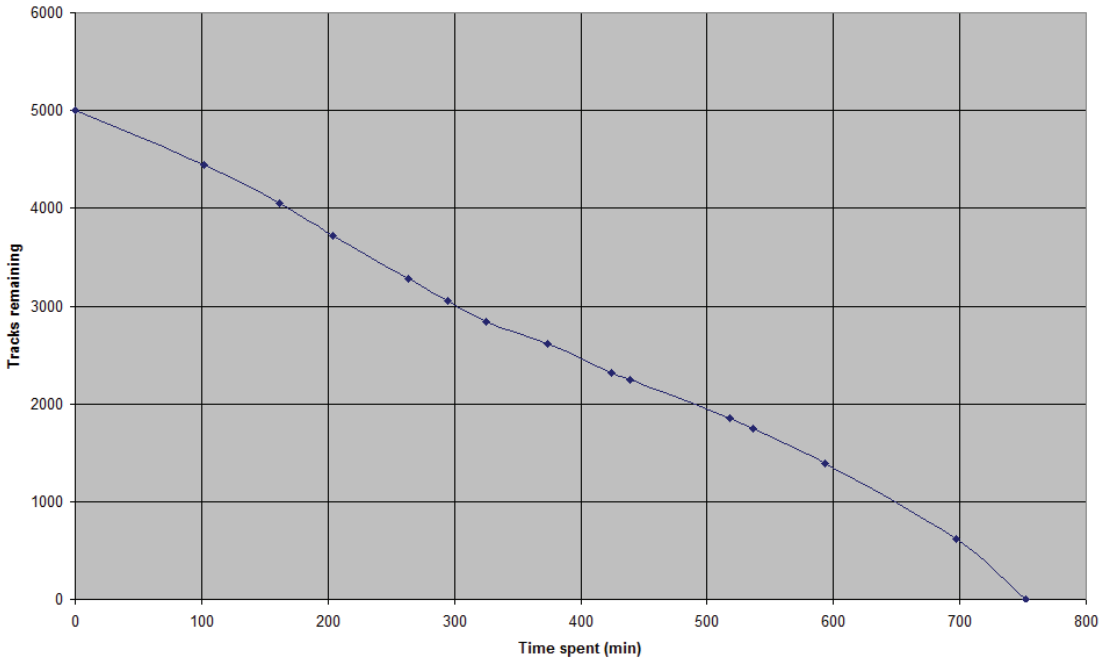

Figure 1: Graph of progress allocating tracks during trial of manual cluster construction

Figure 1 shows a graph of the progress rate from Table 1. The average rate of progress in allocating tracks to clusters or outlier status during the trial was approximately 6.65 tracks per minute. The slope of the graph is fairly consistent, indicating that the user interface and workflow enables a user to maintain a steady rate of progress throughout the process of manually constructing the clustering.

Automatic Outlier Detection

The experience from the trial was that the automatic outlier detection feature was vital for enabling the user to efficiently process tracks.

When the user is constructing a cluster, they are trying to obtain a high quality cluster. This means they must obtain a list of tracks that contains all the nearby tracks that should be included in the same cluster as the seed. From the user's perspective, the best way to reliably achieve this is to set the thresholds for searching around the seed track to a conservatively high level. This will ensure that the initial list of candidates suggested by the system will contain all the tracks that should be in the list. Because the thresholds are set high, the list will also contain false positive results — *i.e.* tracks that should not be included. However the user can remove these false positives from the list. Once the user has done this, they will obtain a list with no false positives and also no false negatives — *i.e.* a high-quality cluster.

Note how the efficiency of this process is heavily influenced by the number of false positive tracks that the system initially proposes and the user must remove before creating the cluster. The false positives included by the system will be comprised of (a) tracks that belong in other clusters, and (b) outliers. Thus reducing the number of outliers in the pool of unallocated tracks will reduce the potential for false positives when searching near a seed track, and hence will generally increase the rate of progress by the user in constructing clusters.

Playback visualisation feature

The experience from the trial was that the playback feature of the user interface was essential for ensuring high-quality clusters were constructed.

The playback feature enables the user to visualise and understand the dynamic behaviour of tracks. This is vital for assessing the similarity of a group of tracks. For example, consider the case of two tracks that follow the same flight path, but at different speeds. On the static map display there will be no indication to the user that the tracks are different. However during a playback of the tracks the difference in speed will become immediately apparent to the user.

It is important note in this regard is that the user interface design for the playback feature (as shown in Figure 4) is suboptimal. To obtain the playback visualisation of tracks, the user must explicitly request the playback controls which appear in a pop-up window which floats above and stays on top of the main interface window. The user closes this pop-up window to stop the playback visualisation. The experience of the trial made it clear that the interface design would be improved if the playback controls were permanently displayed in the main interface window. A toggle button would allow the user to activate or deactivate the playback visualisation at will. The play/pause button, the time display, and time selection slider would simply enter an inactive state (*i.e.* appear greyed out and become unresponsive to mouse clicks) when the user is not requesting a playback.

Additional features:

Based on the experience of the trial, some features that are not present in the Blini system but would be desirable were identified.

- *Non-arbitrary seed selection*
  The experience from the trial was that the arbitrary selection of a seed track from the unallocated track pool at the start of the "Search from seed" procedure could be improved. It was found that sometimes the track randomly selected by the system as the seed was a "bad seed", in the sense that searching around that particular seed would not result a coherent group of tracks that formed a high quality cluster. The user is at times forced to instruct the system to abandon the search around the seed that was selected by the system, causing the system to return that track to the pool of unallocated tracks, and try again with a newly (and again arbitrarily) selected seed.
  This suggests that the workflow would be improved by replacing the arbitrary selection of seeds tracks with a heuristic that is more likely to select good seeds and avoid bad seeds. If such a heuristic could be found it would improve the efficiency of the workflow and potentially improve the quality of the clusters the user produces.

- *Support for hierarchical clustering.*
  Some clustering algorithms are designed to produce hierarchical clustering results. In a hierarchical clustering, not only are data elements grouped into clusters, but these clusters are grouped into higher-level "clusters of clusters", which are in turn grouped into "clusters of clusters of clusters", and so on.
  However the interface of the Blini system was not designed to support the manual construction of a clustering with hierarchical structure. Hence the Blini system is unable to support investigation into the quality of the output of hierarchical clustering algorithms.

- *Support incremental construction with respect to new data*
  The Blini system as a whole, and consequently the user interface described in this report, were designed around the assumption of an unchanging test data set. So when an unallocated track is being processed by the user, the system never consults existing clusters to see if the track should be allocated to an existing cluster (since any track that belongs in the cluster was allocated to it at the time the cluster was created). This means that an existing hand-crafted clustering cannot be updated with more data — instead the user is forced to rebuild the entire clustering from scratch. Redesigning the system to support incremental updating of a hand-crafted clustering with additional data would make the system more suitable for migration out of a laboratory environment into a simulated operational environment for further evaluation.

4. References

Jain, A, Murty, M & Flynn, P (1999), 'Data clustering: a review', *ACM Computing Surveys* 31(3).

Han, J & Kamber, M (2006), *Data Mining: Concepts and Techniques*, 2nd ed., Morgan Kaufmann.

Lowry M (2013), *Towards Evaluation of Air Surveillance Track Clustering Algorithms via External Cluster Quality Measures*, Technical Report DSTO-TR-2800, Defence Science and Technology Organisation.

This page is intentionally blank

Appendix A

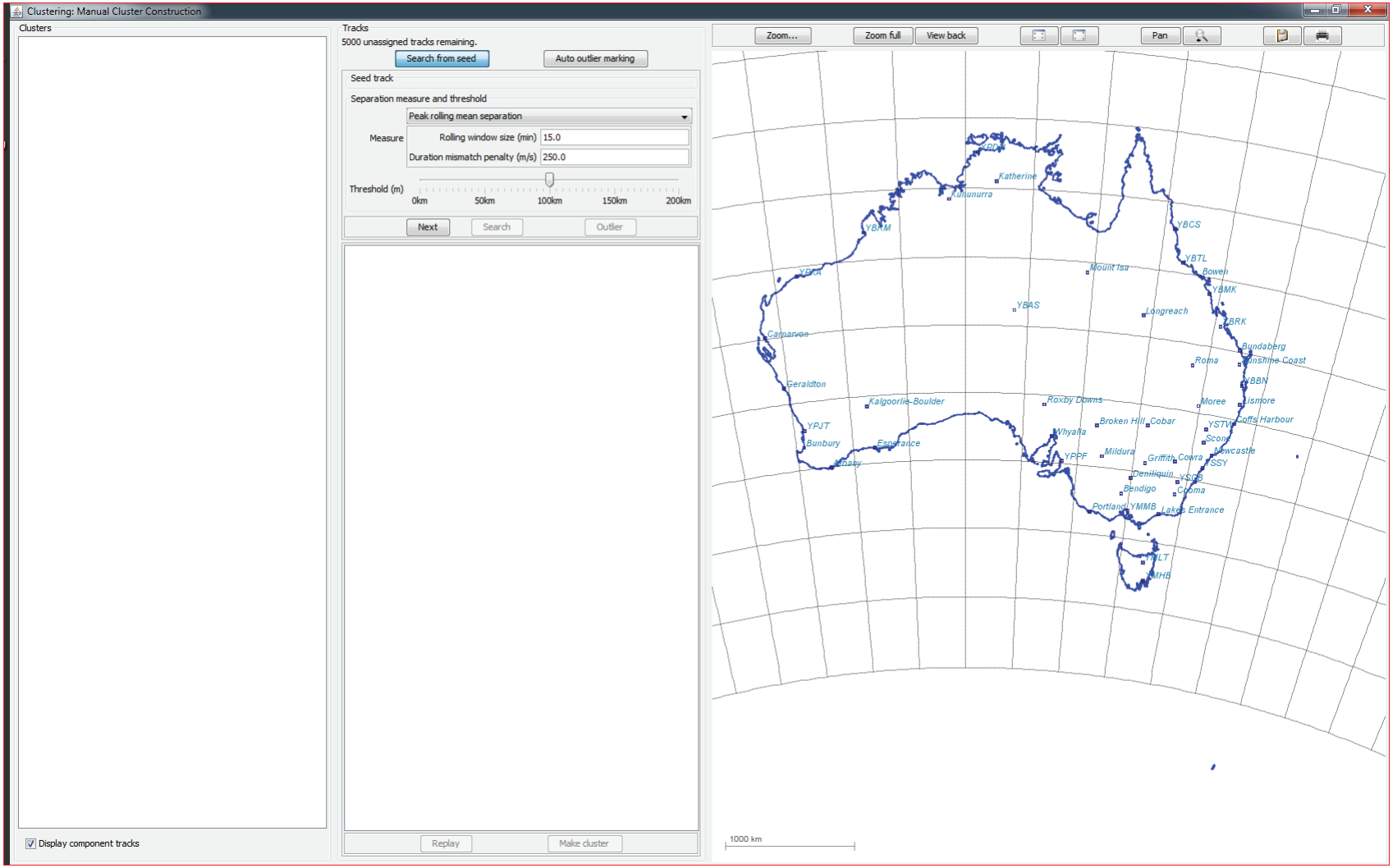

Figure 2: Primary interface for manual clustering construction

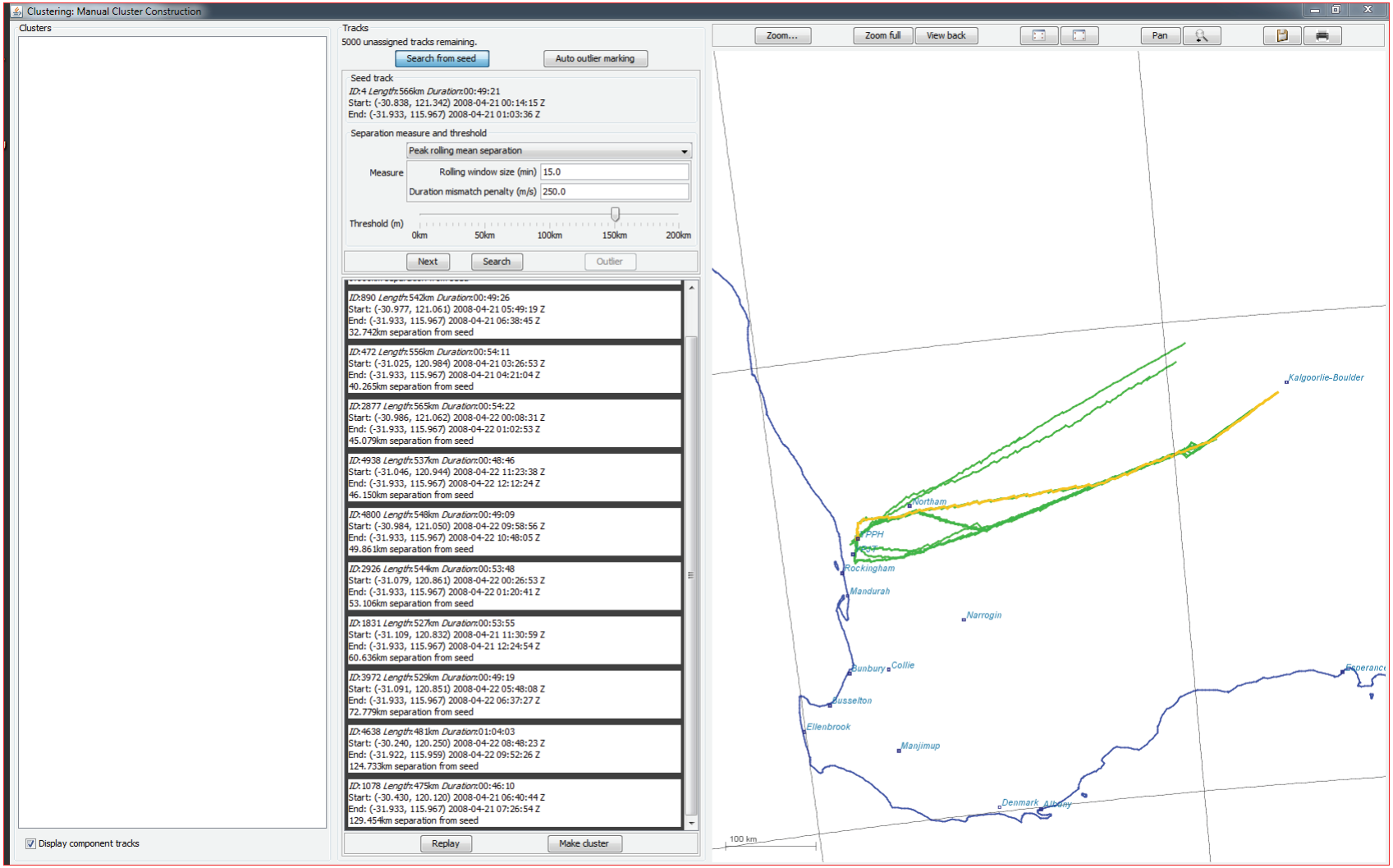

Figure 3: "Search from seed" mode; initial search showing seed track (yellow) and nearby tracks (green)

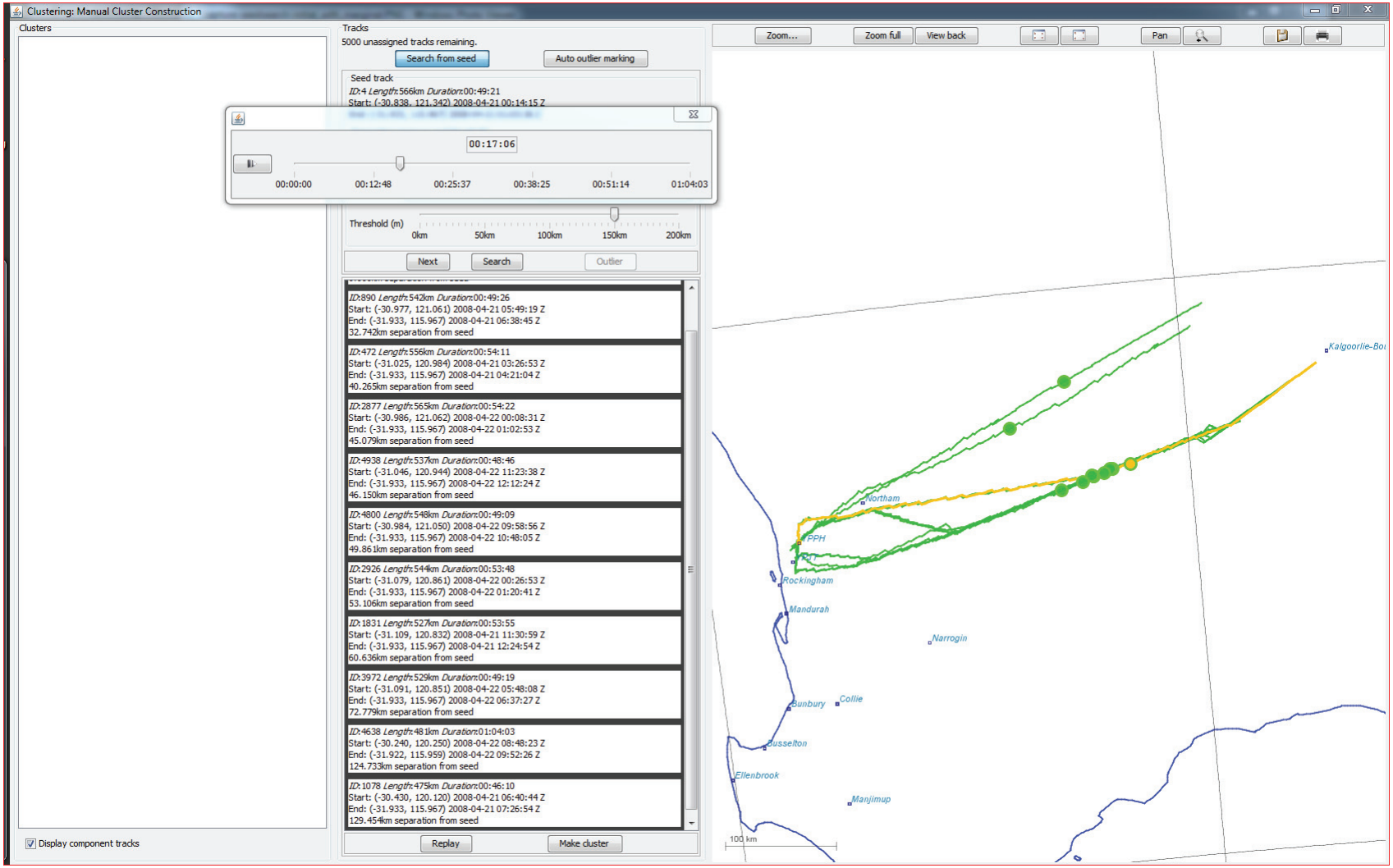

Figure 4: "Search from seed" mode; replay examination of tracks suggested by system for inclusion in a cluster

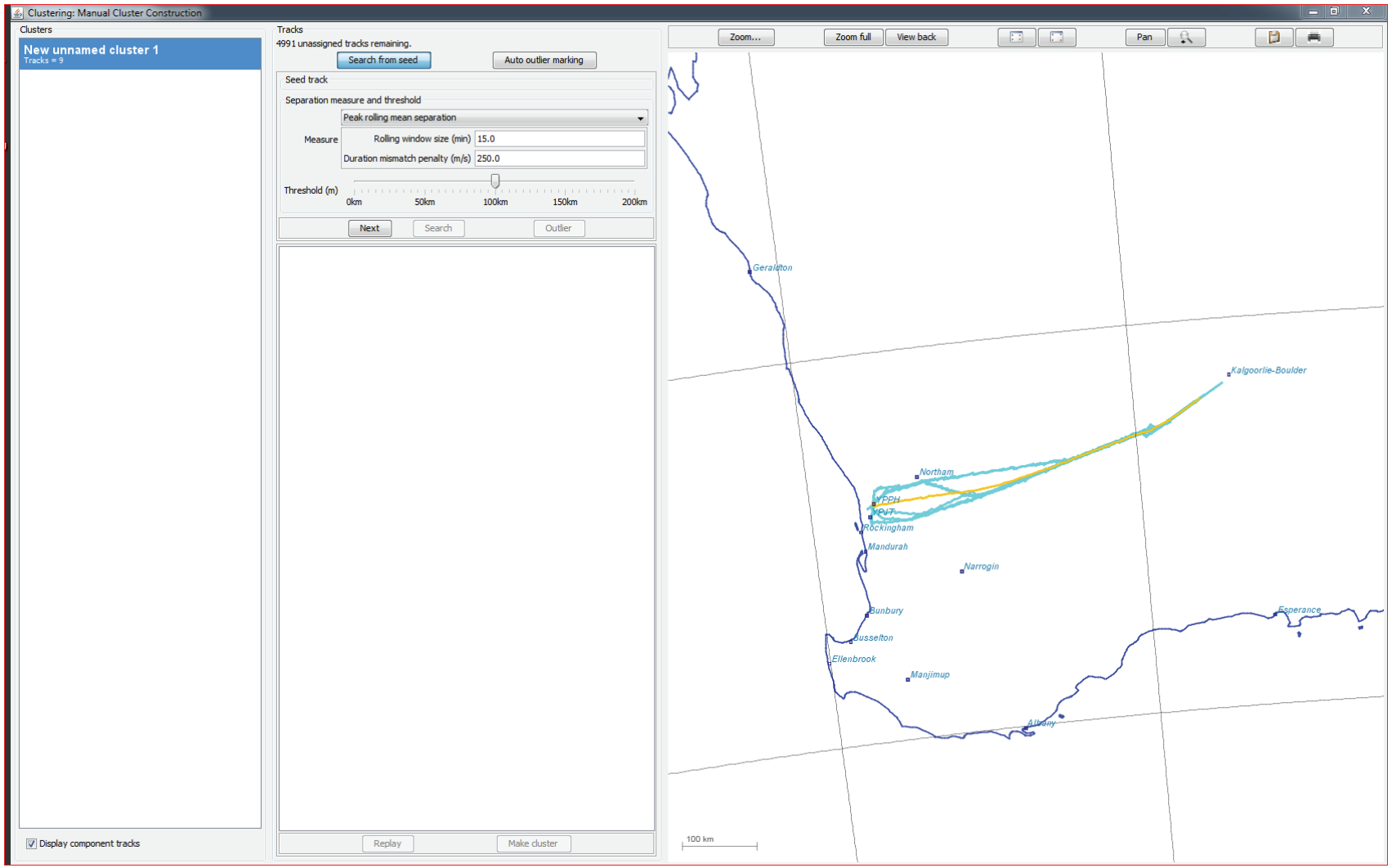

Figure 5: "Search from seed" mode; user has removed inappropriate tracks and created a cluster

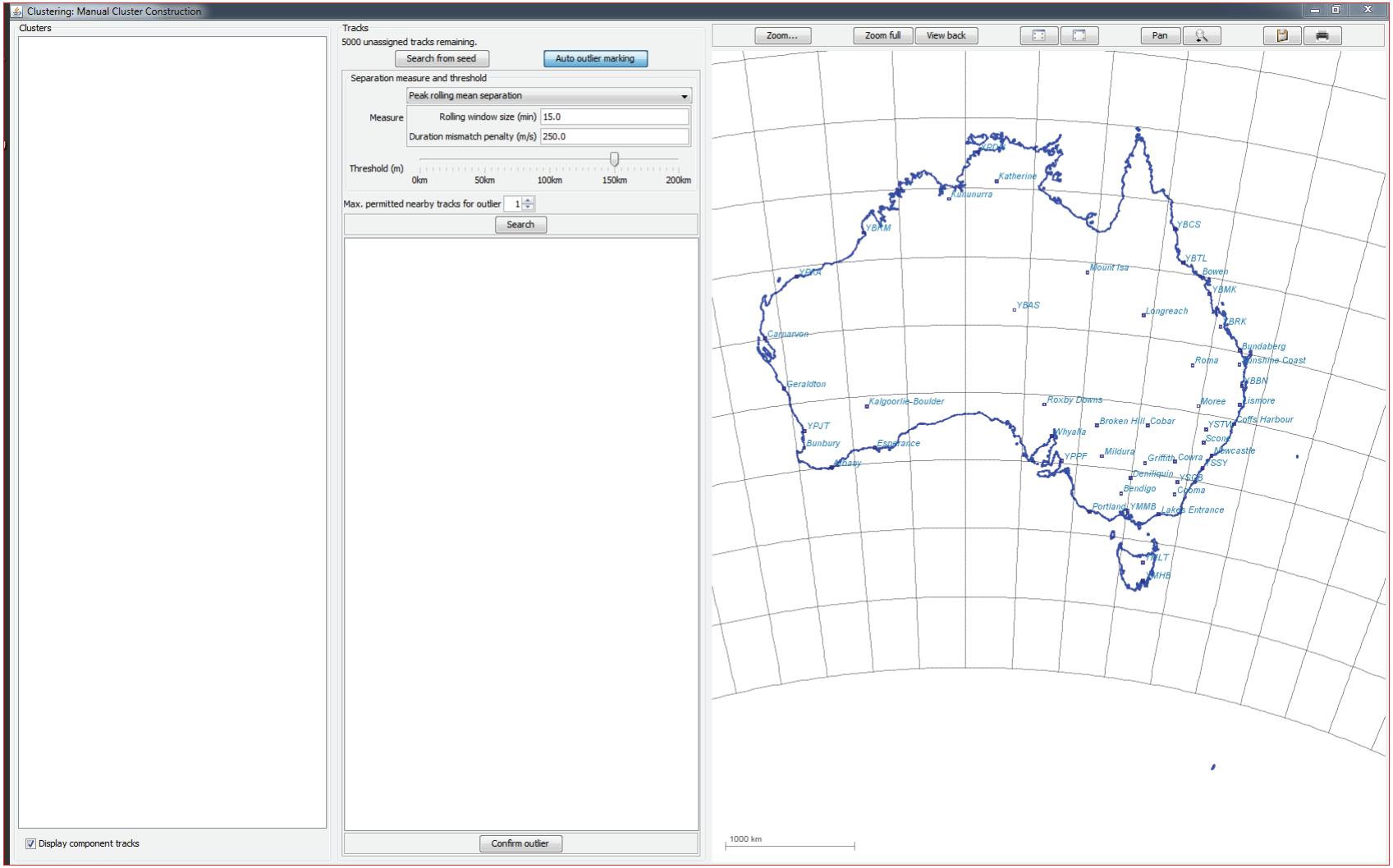

Figure 6: "Auto outlier marking" mode; user selects thresholds for outlier status (at most one other track within 150 km in this example)

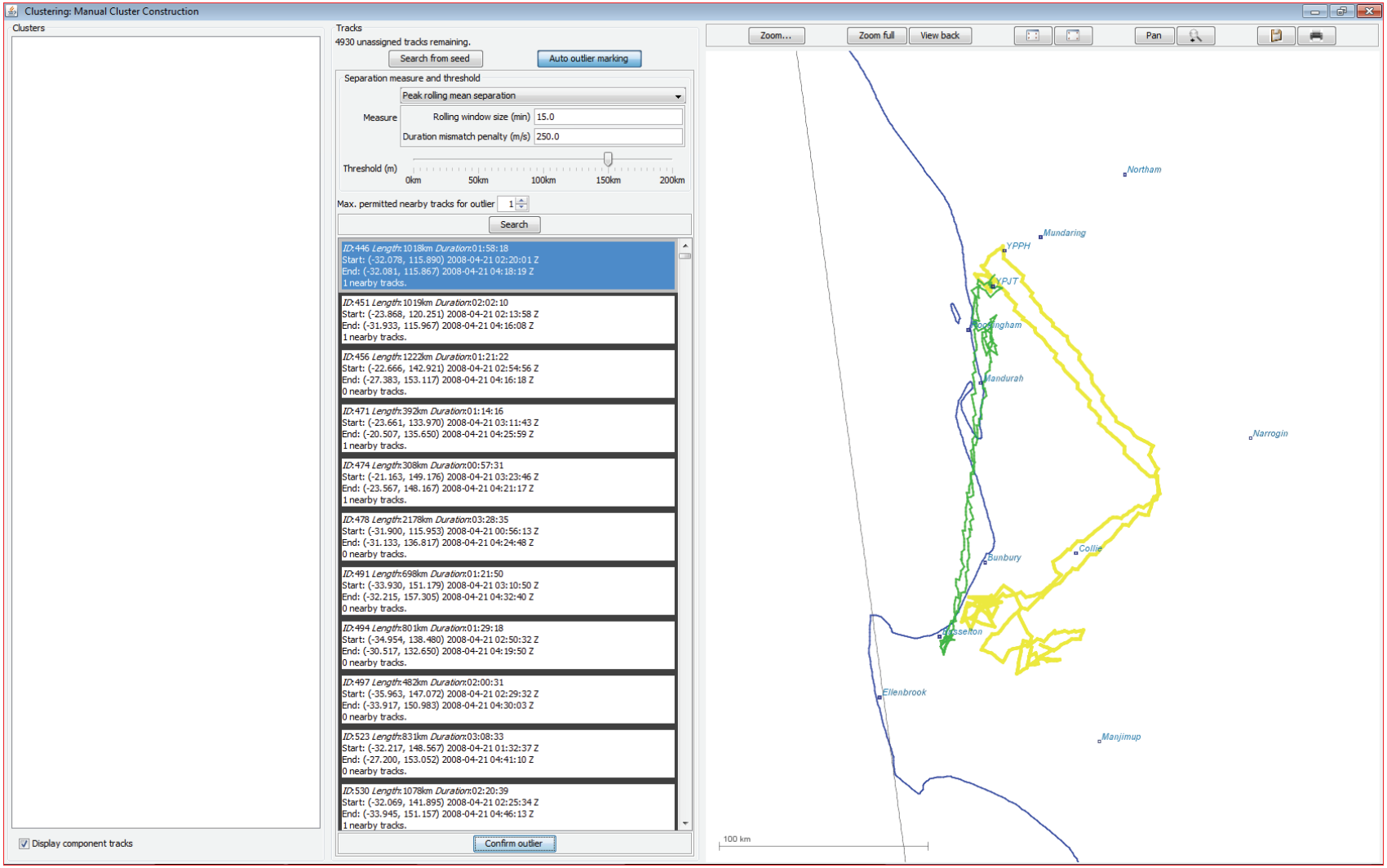

Figure 7: "Auto outlier marking" mode; examining proposed outliers to determine if threshold was sufficiently conservative before mass outlier marking. The user is viewing a suggested outlier (yellow) and the system is showing the only nearby track (green).

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

Experience with a System for Manual Clustering of Air Surveillance Track Data

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

Matthew C. Lowry

5. CORPORATE AUTHOR

DSTO Defence Science and Technology Organisation
PO Box 1500
Edinburgh South Australia 5111 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-GD-0749	AR-015-637	General Document	June 2013

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
2013/1076270/1	INT07/356	DIO	19	3

13. DSTO Publications Repository

http://dspace.dsto.defence.gov.au/dspace/

14. RELEASE AUTHORITY

Chief, Command, Control, Communications and Intelligence Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DSTO RESEARCH LIBRARY THESAURUS

Data Mining, Cluster Analysis, Air Surveillance

19. ABSTRACT

Using clustering algorithms to mine air surveillance track data for groups of similar flights has the potential to facilitate a variety of capability enhancements. Since there are many algorithms that could be used, a method for assessing the quality of algorithm output is required. One potential method is to have a human expert hand-craft a clustering for a test data set, and use this manual clustering as the gold-standard against which the output of a clustering algorithm is assessed. For complex spatio-temporal data such as air surveillance track data, the manual construction of clusterings for a robust test data suite will be labour-intensive and reliant on good information technology support. This report describes an experimental system providing a user interface and workflow for performing manual clustering of air surveillance track data, and experience with a trial of the system.